Device Class Power Management Reference Specification

Network Device Class

This specification is co-authored by Microsoft and
Advanced Micro Devices, Inc.

Draft proposal
v0.7
Send comments to power@microsoft.com

Table of Contents

Revision History

Revision	Date	Comments
0.7	9/23/96	Incorporates 9/20 feedback
0.6	6/14/96	Incorporates 6/11 feedback
0.3	6/3/96	Third edit after 6/3 meeting
0.2	6/3/96	Second edit before 6/3 meeting
0.0	5/1/96	Initial proposal for consideration

Scope

This specification defines the behavior of network devices as it relates to power management, and, specifically, to the four device power states defined for the OnNow Architecture. This specification applies to network devices. It is intended that network device vendors and system makers will be able to design consistent power-manageable products, and that OS vendors will be able to implement an appropriate network device power management policy based on the contents of this specification.

General Device Power Management Considerations

In the OnNow architecture, power management of individual devices is the responsibility of a policy owner in the Operating System, generally a class-specific driver. This policy-owner will implement a power conservation policy that is appropriate for devices in its class. The policy will operate in conjunction with a global system power policy implemented in the operating system (i.e. is the system Working or Sleeping?). In general, the device-class power conservation policy strives to reduce power consumption while the system is Working by transitioning amongst various available power states according to device usage. Since the policy-owner in the Operating System has very specific knowledge of when a device is in use, or potentially in use, there is no need for hardware timers or such to determine when to make these transitions. Similarly, this level of understanding of device usage makes it possible to use fewer device power states. Generally, intermediate states attempt to draw a compromise between latency and consumption due to the uncertainty of actual device usage. With the increased knowledge in the OS, crisp decisions can be made about whether the device is needed at all. With this ability to turn devices off more frequently, the benefit of having intermediate states diminishes.

The policy-owner also determines what class-specific events can cause the system to transition from Sleeping to Working, and enables this functionality based on application or user requests. Note that the definition of the wake-up events that each class supports will influence the system's global power policy in terms of the level of power conservation the Sleeping state can attain while still meeting wake-up latency requirements set by applications or the user.

In the OnNow architecture, bus drivers also implement power policy for their bus class (e.g. PCI, USB, etc..). In general, the Bus driver has responsibility for tracking the device power states of all devices on its bus, and transitioning the Bus itself to only those power states that are consistent with those of its devices. This means that the Bus state can be no lower than the highest state of one of it's devices. However, enabled wake-up events can affect this as well. For example if a particular device is in the D2 state and set to wake-up the system, and the bus can only forward wake-up requests while in the D1 state, then the Bus must remain in the D1 state even if all devices are in a lower state.

Device power state transitions are explicitly commanded by the driver and invoked through bus-specific mechanisms (e.g. ATA Standby command, USB Suspend, etc..). In some cases, bus-specific mechanisms are not available and device-specific mechanisms must be used. Note that the explicit command for entering the D3 state may be the removal of power.

The following definitions apply to devices of all classes:

- **D0**: Device is on and running. It is receiving full power from the system, and is delivering full functionality to the user.
- **D1**: Class-specific low-power state (defined below) in which device context may or may not be lost. Buses in D1 cannot do anything to the bus which would force devices on that bus to loose context.
- **D2**: Class-specific low-power state (defined below) in which device context may or may not be lost. Attains greater power savings than D1. Buses in D2 may cause devices on that bus to loose some context (e.g. the bus reduces power supplied to the bus). Devices in D2 must be prepared for the bus to be in D2 (or higher).
- **D3**: Device is off and not running. Device context is lost. Power may be removed from the device.

Any device context lost must be restored by the device driver when returning the device to the D0 state.

Network Device Power State Definitions

D0

- Physical Layer: On (powered)
- Device is on and running and is delivering full functionality to the user.
- Cable Connect / Disconnect can be detected in this state.
- Network Controller Context: Preserved

D1

- Physical Layer: On (powered)
- Network Controller Context: Preserved
- Receiver Enabled
- Transmitter Disabled (link must be kept up, e.g. 10Base-t must send link pulses, 100Base-TX must send active idles and tokens must be passed for 802.5)
- Cable Connect / Disconnect can be detected in this state.
- Network Wake-up frames can cause wake-up events
- No data transfer to system memory may occur in this state
- The device may discard all frames, including the wake-up frame, after they have been scanned for a wake-up pattern. The device may, optionally, retain the wakeup packet for improved wakeup latency.

D2

This state is not defined for network devices. Use D3.

D3 (Power may be removed)

- Physical Layer: Off
- Network Controller Context: Lost
- Should power be removed, a hardware reset is required when power is re-applied. Upon hardware reset, controller will return to state D0

  Latency to return to D0: Less than 500 ms

Network Device Power Conservation Policy

Present State	Next State	Cause
D0	D1	• Transmit inactivity for a period of time (T1) • System enters sleep state with wake up enabled
D0, D1	D3	• System initiated network shutdown • Cable disconnect • No protocols bound to driver
D1	D0	• Network Wake-up frame received • System initiated network activity
D3	D0	• System initiated network activity
D1, D3	D0	• Hardware Reset

Network Wake-up Events

Wake-up events for the Network device class are defined as:
1) Cable disconnect
2) Network Wakeup Frame received

Network Wakeup Frame Received

The goal is to put the system into a low power state, yet leave the network adapter sufficiently active so that it can issue a wake-up signal when a frame arrives that is interesting for that machine. Interesting frames might be ARP requests, NetBIOS name lookups, any frame directly addressed to the machine, etc. These frames may be addressed as broadcasts, multicasts, or directly addressed frames.

For example, before going into a low power state, the computer MACHINE1 might program the network adapter to wake up the system if it receives any of the following frames:

- NetBIOS lookup of MACHINE1 on NetBEUI
- NetBIOS lookup of MACHINE1 on TCP
- NetBIOS lookup of MACHINE1 on IPX
- Lookup of MACHINE1 on direct host IPX
- ARP of 155.55.555.55
- Any frame with my MAC address in the destination address field

If the network adapter sees any of these frames, then it shall cause a bus specific wake up signal to be asserted.

Some fields in each frame will vary. For example, each of these frames invariably has a source address field which is not necessarily of interest to the decision to wake up the system. Since we don't want to look at every byte in an incoming frame, for each sample wake-up frame we can provide a byte mask that indicates which bytes of the incoming frame should be ignored.

In addition to sample frames and byte masks, the network adapter can also consider the address type when looking for a wake-up frame. The address type is used in two different ways. First the network adapter maintains three Global Accept Bits that determine whether the adapter should wake up the system on receipt of any unicast, multicast, or broadcast frame that would be received in the normal operating mode. Second, along with each sample wake-up frame and byte mask is a three-bit address type field that

specifies whether the sample frame should be compared with unicast, multicast, or broadcast frames or with some combination such as multicast and broadcast frames.

The three Global Accept Bits are:
1. Global Unicast Accept. When this bit is set, any frame addressed to the physical address of this node will cause a wake-up event.
2. Global Multicast Accept. When this bit is set, any frame addressed to a multicast address in this node's multicast address table will cause a wake-up event.
3. Global Broadcast Accept. When this bit is set, any frame addressed to the broadcast address will cause a wake-up event.

Since the decision to assert the wake-up signal is usually based on information from the frame header, it is not necessary for the device to scan the entire frame or to store byte masks long enough to cover a maximum length frame. Also some devices may not be able to do multicast filtering when they are not in the D0 state. Therefore the system software must be able to query the adapter's driver to determine
1. how many sample frames the adapter can accept,
2. the maximum byte mask length that it can accept, and
3. whether or not the device can filter multicast frames in the D1 state.

The system software must make sure that it does not attempt to exceed these capacities.

Before putting the network adapter into the wake-up state, the system passes to the adapter's driver a list of sample frames, corresponding byte masks, and corresponding address type fields. Each sample frame is an example of a frame that should wake up the system. Each byte mask defines which bytes of the incoming frames should be compared with the corresponding sample frame in order to determine whether or not to accept the incoming frame as a wake-up event.

Each address type field contains three bits:
1. Unicast. When this bit is set, all frames addressed to the physical address of this node will be compared with the sample frame.
2. Multicast. When this bit is set, all frames addressed to a multicast address in this node's multicast address table will be compared with the sample frame.
3. Broadcast. When this bit is set, all frames addressed to the broadcast address will be compared with the sample frame.

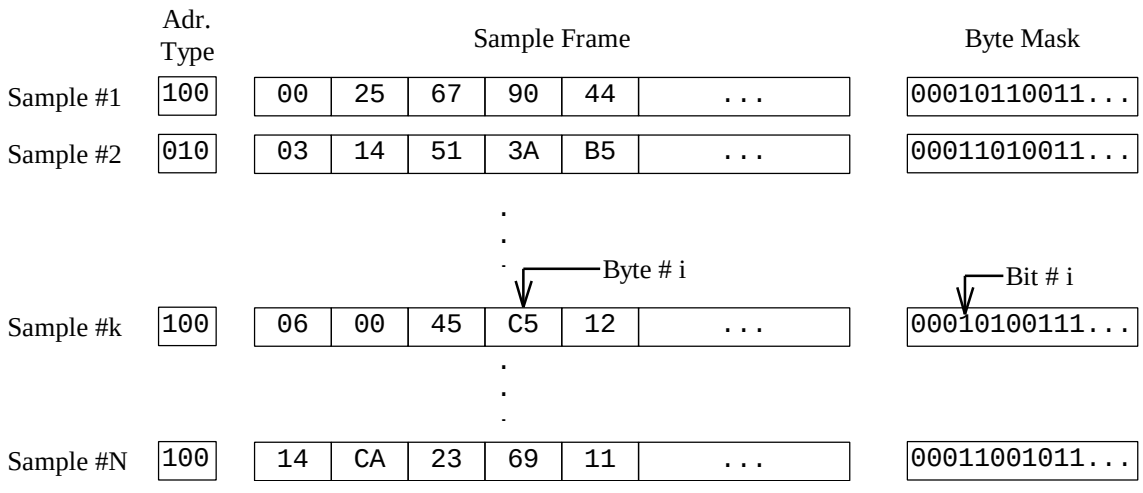

Figure 1. A List of Sample Frames

When the network adapter is operating in the wake-up mode, the byte masks work as follows: If bit number *i* of byte mask number k is 1, then byte number *i* of each incoming frame will be compared with byte number *i* of sample frame number k. Otherwise byte number *i* of sample frame number k will be ignored in the wake-up comparison. If all of the bytes thus selected from one of the sample frames match the corresponding bytes of the incoming frame, and the address type of the incoming frame matches one of the address types that are enabled in the address type field of the template, then the incoming frame will cause the network adapter to generate a bus specific wake-up signal.

The first bit of the byte mask corresponds to the first byte of the frame. For example, for an Ethernet or 802.3 frame the first bit of the byte mask corresponds to the first byte of the Destination Address Field of the Ethernet or 802.3 header.

(Note that pattern or byte mask storage space could be reduced at the expense of additional hardware complexity if the driver would store a pointer to the first byte of the frame that should be examined. For example if the first 30 bytes of the frame are to be ignored, the driver could store the number 30 along with a sample frame that is 30 bytes shorter than the original sample frame that the operating system gave to the driver. For a device that is built with this type of interface, the software driver is responsible for examining the parameters passed from the operating system to determine the offset and to truncate the sample frames appropriately.)

For each sample wake-up frame the operating system must pass to the network adapter driver the following information before the adapter goes into the wake-up mode:

- The contents of each sample frame.
- A byte mask that tells the driver which bytes of the incoming data stream to accept.
- The three-bit address type field for the frame.

In addition the operating system must pass to the driver adapter the three Global Wake-up Address Type bits.

When the system is in low power mode, the network adapter will listen as it does in normal operation. When it gets a frame, it will compare that frame against the list of frames the network software gave it.

If the incoming frame is directly addressed to this adapter, and the Global Unicast Accept bit is set, then the network adapter shall cause a bus specific wake up event. If the incoming frame is multicast and is directed to an address in the adapter's multicast address table, and the Global Multicast Accept bit is set, then the network adapter shall cause a bus specific wake up event. If the incoming frame is a broadcast, and the Global Broadcast Accept bit is set, then the network adapter shall cause a bus specific wake up event. The above is true regardless of the contents of the frame, or the programming of the match patterns and mask bits.

In the example below since the incoming frame passes the comparison, and has a unicast address, and the unicast bit is set for that frame pattern, the frame would be a wake-up frame. Had the unicast been cleared in this example, this frame would not be a wake-up frame.

```
Sample frame address type field:          U  M  B
                                          1  0  0

Sample frame:          66 AA 00 04 05 06 07 00 BB 00
Mask for above frame:   0  0  0  1  1  1  1  0  0  0
Incoming frame:        10 02 03 04 05 06 07 08 09 0A
```

Figure 2. A Sample Frame and Byte Mask

If one of bytes 4-7 had been different, the comparison would have failed and the network adapter would move on to the next item in the list. If the network adapter goes through all the frames in the list and cannot find a match, it simply discards the frame.

Appendix A includes samples of wake-up packets that might be used for a typical protocol.

Operating System Requirements

In order to accommodate network wake-up events, the operating system must provide a standard interface for the following functions:

- The system software must query the driver to determine how many sample frames are supported by the hardware (minimum 4). The system software must make sure that it does not attempt to load more patterns into the device than the device can accept.
- The system software must also query the driver to determine the maximum number of bytes of an incoming frame that can be compared with stored patterns. The system must make sure that it does not attempt to load patterns into the device that are longer than the device can accept.
- The system software must be able to determine whether or not the device can filter multicast addresses while it is in the D1 state.
- The system software must pass the Global Accept Bits to the driver.
- For each wake-up pattern the system software must pass to the driver
  1. The contents of each sample frame.
  2. A byte mask that tells the driver which bytes of the incoming data stream to accept.
  3. The three-bit address type field for the frame.

Exact vs. Signature Matching

The above description specifies perfect matching of the incoming data with selected bytes of stored frames. This may be undesirable or even impossible, depending on the architecture of the network adapter. While perfect matching would be the best solution, other mechanisms, such as hashing or check summing the incoming frame may offer adequate performance while reducing the hardware requirements.

For example instead of storing complete patterns, the device can store only a four-byte CRC value and a byte mask for each sample frame. This approach requires one CRC generator circuit for each sample frame. For each sample frame and byte mask that the system software passes to the driver, the driver calculates a CRC value based on those bytes of the sample frame whose corresponding mask is set to 1. The driver stores the resulting value and corresponding byte mask in the controller hardware. When wake-up mode is enabled, as a frame arrives from the network, each CRC generator calculates a CRC value based on those bytes of the incoming frame whose corresponding masks are set to 1. If for any of the CRC generators, the calculated value matches the stored value and if the incoming frame passes the standard CRC check, the device asserts the wake-up signal.

For the example shown in Figure 2, the network driver computes a 32-bit CRC based on the 4 bytes of the frame with mask set to 1, namely bytes 4, 5, 6, and 7. The network driver loads this CRC value and the byte mask into the network adapter. When a frame is being received, the network adapter calculates a

CRC value using bytes number 4, 5, 6 and 7 of the incoming frame. When the incoming frame ends, the calculated CRC is compared to stored CRC. If there is a match, and if the whole frame's CRC is also correct, a wake-up event will be generated.

Other signature matching implementations are possible. The only requirement is that the scheme must generate no more than 1 false wake-up for every TBD number of incoming frames.

Compatibility with Other Power Management Mechanisms

The power management scheme described here must be compatible with older schemes such as Magic Packet™ Technology. In fact both mechanisms can be implemented in the same network device. Both mechanisms can drive the same wake-up signal and both can be enabled at the same time. The other wake-up mechanism can be controlled by software calls that are not defined in this specification.

For example Magic Packet frame detection can be implemented in a manner that is both compatible with On Now software and at the same time independent of the operating system and the protocol stack. Most network devices use an EEPROM configuration memory to store the IEEE MAC address. A Magic Packet control bit could be defined in this EEPROM. During the BIOS initialization time, the network device hardware or BIOS software could enable or disable the Magic Packet option based on the state of this Magic Packet control bit.

Appendix A: Example Wakeup Frames

Below are the 9 wakeup frames defined for a default installation of a Windows machine running only TCP/IP. Note that each running protocol would have it's own set of wakeup frames. An example for a multi-protocol machine will be added for the final version of the specification.

Note: 'computername' in all below is "WAKER"; logged on user name in all below is "WAKEMAN"

ARP to machine address 157.55.199.72

Offset: Hex Pattern:
==========
12 0806
21 01
38 9d37c748

NBT Name Query for computername <00>

Offset: Hex Pattern:
==========
12 0800
23 11
37 8900
44 0110
54 20 46 48 45 42 45 4c 45 46 46 43 43 41 43 41 43 41 43 41 43 41 43 41 43 41 43 41 43 41 41 41

NBT Name Query for computername <03>

Offset: Hex Pattern:
==========
12 0800
23 11
37 8900
44 0110
54 20 46 48 45 42 45 4c 45 46 46 43 43 41 43 41 43 41 43 41 43 41 43 41 43 41 43 41 43 41 41 44

NBT Name Query for computername <20>

Offset: Hex Pattern:
==========
12 0800
23 11
37 8900
44 0110
54 20 46 48 45 42 45 4c 45 46 46 43 43 41 43 41 43 41 43 41 43 41 43 41 43 41 43 41 43 41 43 41

NBT Name Registration for computername <00>

Offset: Hex Pattern:
==========
12 0800
23 11
37 8900
44 2910
54 20 46 48 45 42 45 4c 45 46 46 43 43 41 43 41 43 41 43 41 43 41 43 41 43 41 43 41 43 41 41 41

NBT Name Registration for computername <03>

Offset: Hex Pattern:
==========
12 0800
23 11
37 8900
44 2910
54 20 46 48 45 42 45 4c 45 46 46 43 43 41 43 41 43 41 43 41 43 41 43 41 43 41 43 41 43 41 41 44

NBT Name Registration for computername <20>

Offset: Hex Pattern:
==========
12 0800
23 11
37 8900
44 2910
54 20 46 48 45 42 45 4c 45 46 46 43 43 41 43 41 43 41 43 41 43 41 43 41 43 41 43 41 43 41 43 41

NBT Name Query for logged on user name <03>

Offset: Hex Pattern:
==========
12 0800
23 11
37 8900
44 2910
54 20 46 48 45 42 45 4c 45 46 45 4e 45 42 45 4f 43 41 43 41 43 41 43 41 43 41 43 41 43 41 41 44

NBT Name Registration for logged on user name <03>

Offset: Hex Pattern:
==========
12 0800
23 11
37 8900
44 2910
54 20 46 48 45 42 45 4c 45 46 45 4e 45 42 45 4f 43 41 43 41 43 41 43 41 43 41 43 41 43 41 41 44